International Journal of Information & Computation Technology.
ISSN 0974-2239 Volume 4, Number 13 (2014), pp. 1263-1273

http://www. irphouse.com

ABC Based Multi-Objective Approach for NoC Design Space Exploration

R K Jena

Institute of Management Technology, Nagpur, India

Abstract

Network-on-Chip (NoC) has recently emerged as an efficient communication solution for the System-on-Chip (SoC) design. Design space exploration and performance evaluation are the most essential task in NoC design. In this paper, an ABC based design space exploration framework for the NoC design is proposed. The objective of the design space exploration is to minimize the total energy consumption and maximum link bandwidth. This research has considered many-many mapping between router and IPs instead of one-one mapping. The results show that the proposed framework saves around 40-50% of energy and around 10% of link bandwidth in compare to other contemporary techniques.

Keywords: NoC, Multi-objective ABC, Analytical Model, Design Space Exploration.

1. Introduction

As technology scales up and chip integrity grows, on-chip communication is playing an dominant role in System-on-Chip (SoC) design. Networks-on-Chip(NoC) is considered as a subset of System-on-Chip (SoC), which combines the state-of-the-art techniques in computer network with that in chip design[1-4]. It has been regarded as a promising technique to solve the bus-based structure issues. The NoC approach has

been proposed as a promising solution to complex intra-SoC communication problems. Compared to traditional bus interconnection architecture, NoC is much more extensible and parallelizable. But, the NoC based system requires a reliable methodology for better design space exploration. The success of these methodologies depends on the availability of adequate performance analysis tools that can efficiently guide the design space exploration. The Figure 1 shows the proposed frame work for the integrated design space exploration at system level. The inputs to the frame work are Noc architecture and the application in the form of task graph. Following the input, the next step is the mapping of application to the architecture. The mapping provides a optimal solution to the design.

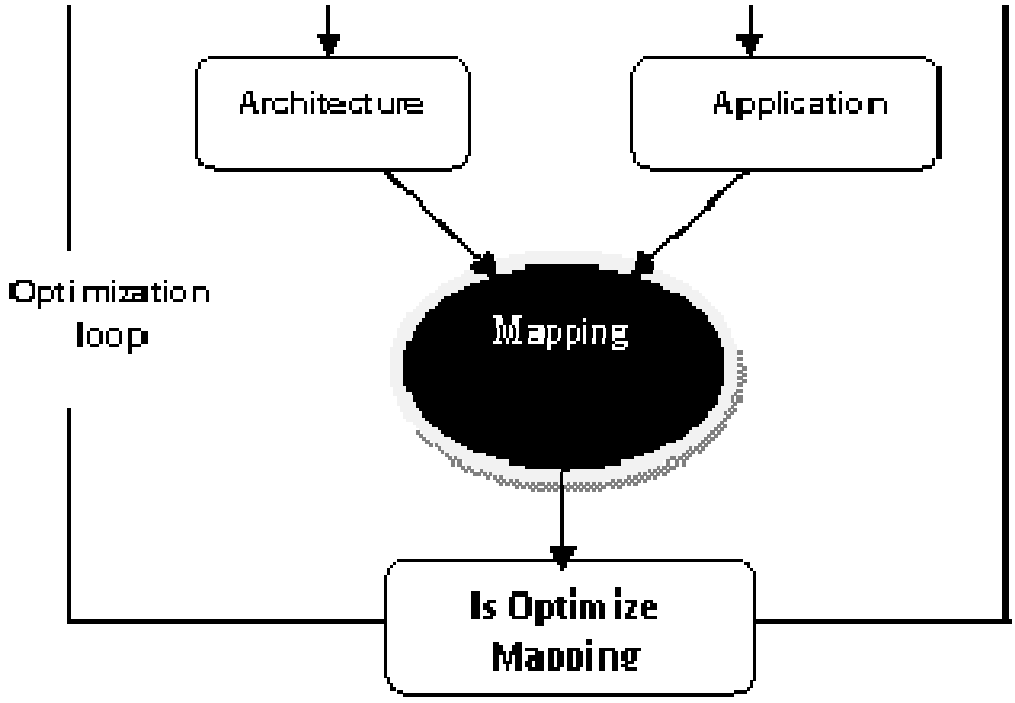

Figure 1: Proposed framework for design space exploration

The remaining of this paper is organized as follows. Section 2 reviews related work and highlights our contributions. Section 3 discusses our proposed design space exploration framework, which uses the switch/router model of the network performance analysis. Experimental results are discussed in Section 4. Finally, the paper is concluded in Section 5 by summarizing the contribution.

Related Work

The design space exploration of NoCs is commonly formulated as a constrained optimization problem [5,6,7]. There are different issues for design space exploration problems like: structure of switch (router), types of mappings between cores (IPs) and switches (routers), buffer space allocation etc. The type of mappings between cores and switches also plays an important role in performance of NoC design. The design of regular NoC architecture has been proposed in past by different researchers [8,9]. Dally et al [8] use the on-chip network instead of adhoc global wiring to interconnect the entire IP module on a chip. Kumar et al [9] implement a 2D mesh NoC architecture that consists of a mesh of IP cores. There is an associated router for each core, and each router is thereby connected to four neighbor routers. Various approaches have been reported for solving the topology selection and NoC mapping problem [10,11]. Lei et al. [10] proposed a two-step genetic algorithm that finds a mapping of cores onto NoC architecture such that the overall execution time is minimized. K Srinivasan et.al [11] introduces a linear-programming based NoC synthesis method. Krishnan Srinivasa [12] utilized the GA for the custom NoC mapping process. Many researchers also introduced mapping and routing methods for the customized NoC[13]. However, firstly, no methodologies involve the design of link-load balance in the course of mapping and routing , and secondly all previous research considered one-one mapping between core and switch (router). So, in this paper, we propose a ABC based design methodology to minimize the energy consumption of NoC while guaranteeing the balance of link-load using many-many mapping between core and switch.

Design Space Exploration Methodology

The main task of the proposed framework is deals with the mapping of the target application onto the NoC architecture. Before describing the mapping tasks in details, the characterization of the application and architecture, which is given as inputs to the proposed framework, is presented below. The application in the proposed framework is represented using task graph. The task graph (core communication graph) is a directed graph G (Vertex(V), Edge(E)) with each vertex $v_i \in V$, representing a task and the directed edge (v_i , v_j), denoted as $e_{ij} \in E$, representing the communication dependency between the cores v_i and v_j . Each edge (e_{ij}) has two variables(V_{ij} and D_{ij}) associated with it. Where V_{ij} represents the required bandwidth and D_{ij} represents

delay. Each task graph is also associated with another constraint (d), which represent the execution deadline of the task graph.

The NoC architecture of the proposed framework is represented by the NoC topology graph (Architecture characterization graph). The NoC topology graph is a directed graph G(Tile(T), Route(R)).Where, each vertex t_i ЄT, representing a tile (node) in the topology and the directed edge (t_i , t_j), denoted as , r_{ij} ЄR¸ representing a directed communication between the vertices t_i and t_j . The weight of each edge (r_{ij}) is denoted by the order pair <$d_{ij,}$ b_{ij}>. where d_{ij} represents the delay and b_{ij} represents the bandwidth of the communication from t_i to t_j. The size of the NoC architecture in the proposed framework is decided accordingly to the size of the task graph. It is very important to choose a proper size of the architecture, since it may lead to significant area overhead if the size of the architecture is not chosen properly. So, in order to map all the functioning blocks into the NoC architecture, the following condition should be satisfied. i.e *Total number of vertices ($|V|$) in the task graph ≤ Total number of vertices ($|T|$) in the NoC topology graph.* After application and architecture specification, the next step of our framework flow is the mapping of the task graph onto the topology graph. The details of the mapping step are described next.

Objective Functions

In this paper, two objective functions i.e the energy consumption and maximum link bandwidth are considered to optimize in the proposed design space exploration. The energy consumption is defined as follows: The energy $E_{bit}^{p(t_{v_i},t_{vj})}$ that every bit route through $P\left(t_{v_i}, t_{v_j}\right)$:

$$E_{bit}^{p(t_{v_i},t_{v_j})} = (|p(t_{v_i},t_{v_j})|+1)\times E_R + |p(t_{v_i},t_{v_j})|\times E_L \quad (1)$$

Where $P(t_{v_i}, t_{v_j})$ indicate the set of routing paths (P) that we map core v_i onto tile 'i' and core v_j onto tile j. It is a set of several links {l_j, l_{j+1}, . . . }. Where $|P(t_{v_i}, t_{v_j})|$ is the hop of path $P(t_{v_i}, t_{v_j})$, and E_R is the energy consumption of single router, E_L is the energy consumption of one link.

Then the energy $E_{e_{ij}}$ that a communication edge routes though the path $P(t_{v_i}, t_{v_j})$:

$$E_{e_{i,j}} = v(e_{i,j}) \times \{(|p(t_{v_i}, t_{v_j})| + 1) \times E_R + |p(t_{v_i}, t_{v_j})| \times E_L\} \quad (2)$$

For the whole application, the energy consumption can be defined as :

$$E_{E_C} = \sum_{e_{i,j} \in E_C} v(e_{i,j}) \times \{(|p(t_{v_i}, t_{v_j})| + 1) \times E_R + (|p(t_{v_i}, t_{v_j})|) \times E_L\} \quad (3)$$

Again, the link-load function is defined as follows:

$$U(l_j) = \sum_{\forall e_{i,j}} w(e_{i,j}) \times f(l_j, p(t_{v_i}, t_{v_j}) \quad (4)$$

Where

$$f(l_j, p(t_{v_i}, t_{v_j})) = \begin{cases} 0 & l_j \in p(t_{v_i}, t_{v_j}) \\ 1 & l_j \notin p(t_{v_i}, t_{v_j}) \end{cases}$$

Moreover, we use the normalized worst link-load γ to evaluate the link-load-balance. For every core's mapping position (M AP)and its source and destination routing path P set, the normalized worst link-load γ:

$$\gamma(\mathrm{MAP}, \mathrm{P}) = \frac{\max U(l_i), \forall\, i \in L}{2WNK_m} \quad (5)$$

Where the denominator in the (5) is the bisection bandwidth of 2D mesh NoC, W is the link bandwidth, N is the node number of NoC, and K_m is the max radix of 2D mesh.

So, finding the mapping instances s.t total energy consumption (excluding IP core energy) is minimized and the value of $\gamma(\mathrm{MAP}, \mathrm{P})$ is minimized i.e

Min {MAP, E_{E_C}} (6)

Min {Max(γ(MAP, P)} (7)

Such that:

$$\forall v_i \in \mathbf{V}, \quad MAP(v_i) \in T \tag{8}$$

$$\forall v_i \neq v_i \in \mathbf{V}, \quad MAP(v_i) \neq MAP(v_i) \tag{9}$$

$$\forall\ l_k, \quad W \geq \sum b(v_{i,j}) \times f(l_k, p(t_{v_i}, t_{v_j})) \tag{10}$$

Artificial Bee Colony (ABC) Algorithm

The artificial bee colony algorithm is a new population-based meta-heuristic approach, initially proposed by Karaboga[14] and further developed by Karaboga and Akay[15] . It has been used in various complex problems. The algorithm simulates the intelligent foraging behavior of honey bee swarms. The algorithm is very simple and robust. In the ABC algorithm, the colony of artificial bees is classified into three categories: employed bees, onlookers, and scouts. Employed bees are associated with a particular food source that they are currently exploiting or are “employed” at. They carry with them information about this particular source and share the information to onlookers. Onlooker bees are those bees that are waiting on the dance area in the hive for the information to be shared by the employed bees about their food sources and then make decision to choose a food source. A bee carrying out random search is called a scout. In the ABC algorithm, the first half of the colony consists of the employed artificial bees, and the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source has been exhausted by the bees becomes a scout. The position of a food source represents a possible solution to the optimization problem, and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution represented by that food source. Onlookers are placed on the food sources by using a probability-based selection process. As the nectar amount of a food source increases, the probability value with which the food source is preferred by onlookers increases too [14,15] . The main steps of the algorithm are given in Algorithm Basic. [14]

Algorithm Basic ( )

Initialization Phase

Repeat
Employed Bees Phase
Onlooker Bees Phase
Scout Bees Phase
Memorize the best solution achieved so far
Until (Maximum Cycle or Maximum CPU time)

In Initial phase, all the vectors of the population of food are initialized. In Employed Bees Phase Employed bees search for new food sources having more nectar within the neighborhood of the food source They find a neighbour food source and then evaluate its profitability (fitness). In Onlooker Bees Phase, employed bees share their food source information with onlooker bees waiting in the hive and then onlooker bees probabilistically choose their food sources depending on this information. In ABC, an onlooker bee chooses a food source depending on the probability values calculated using the fitness values provided by employed bees. For this purpose, a fitness based selection technique can be used, such as the roulette wheel selection method etc.

NoC Design Space Exploration using Multi-Objective ABC

As opposed to single-objective optimization, Multi-objective Optimization (MO) usually maintain a non-dominated solutions set. In multi-objective optimization, for the absence of preference information, none of the solutions can be said to be better than the others. Therefore, in our algorithm, an External Archive (EA) is created to keep a historical record of the non-dominated vectors found along the search process. In the initialization phase, the external archive will be initialized. After initializing the solutions and calculating the value of every solution they are sorted based on non-domination. Then each solution is compared with every other solution in the population to find which one is non-dominated solution. Then put all non-dominated solutions into the external archive.

In the onlooker bees' phase, a comprehensive new learning strategy is used to produce the new solutions v_i. For each bee x_i, it randomly chooses m dimensions and learns from a non-dominated solution which is randomly selected from EA. The other dimensions learn from the other non-dominated solutions. The new solution is produced by using the following expression:

$$v_{i,f(m)} = x_{i,f(m)} + \emptyset(m)(EA_{k,f(m)} - x_{i,f(m)} \quad (11)$$

where k ϵ (1,2,…,p) is randomly chosen index and p is the number of solution in the EA. f(m) is the first m integers of a random permutation of the integers 1 to n, and f(m) defines, which x_i dimensions should learn from EA_k . $\emptyset(m)$ produce m random numbers which are all between [0,2]. The m random numbers correspond to the m dimensions. Each remaining dimension learns from the other non-dominated solutions by using:

$$v_{ij} = x_{ij} + \emptyset_{ij}(EA_{lj} - x_{ij}) \quad (12)$$

Where

l≠k, j ϵ (1, 2, 3,…,p) and j not ϵ f(m)

The external archive will be updated at each generation. As the evolution progresses, more and more new solutions enter the external archive. Considering that each new solution will be compared with every non-dominated solution in the external archive to decide whether this new solution should stay in the archive. As the computational time is directly proportional to the number of comparisons, the size of external archive must be limited. Finally, if the number of non-dominated solutions exceeds the allocated size of EA, then the crowding distance is used to remove the crowded members crowding distance.

Experimental Result

This section provides a detailed analysis of experimental results of the proposed framework. Three random CCG of different sizes (4,9, 16) are chosen for the experiment. The random CCG is generated with TGFF[26]. There are total 16 nodes in the task graph assign to 16 selected cores (IPs) available from industry. Then these cores are mapped onto 4×4 2D mesh architecture using mapping. For the experimental purpose, the required bandwidth of an edge is uniformly distributed over the range [0,500MB/s] and the traffic volume of an edge is uniformly distributed over the range [0,1GB]. The result of our ABC formulation is denoted as MABC. The results of MABC are compared with ACO based algorithm [2]. The results obtained

by this approach (MABC) (for three synthetic benchmarks) are compared with ACO. Figure-2 shows that MABC saves around 10% of maximum link bandwidth as compare to ACO based algorithm. Figure-3 shows that MPO saves around 50% of energy consumption in compare to ACO based algorithm.

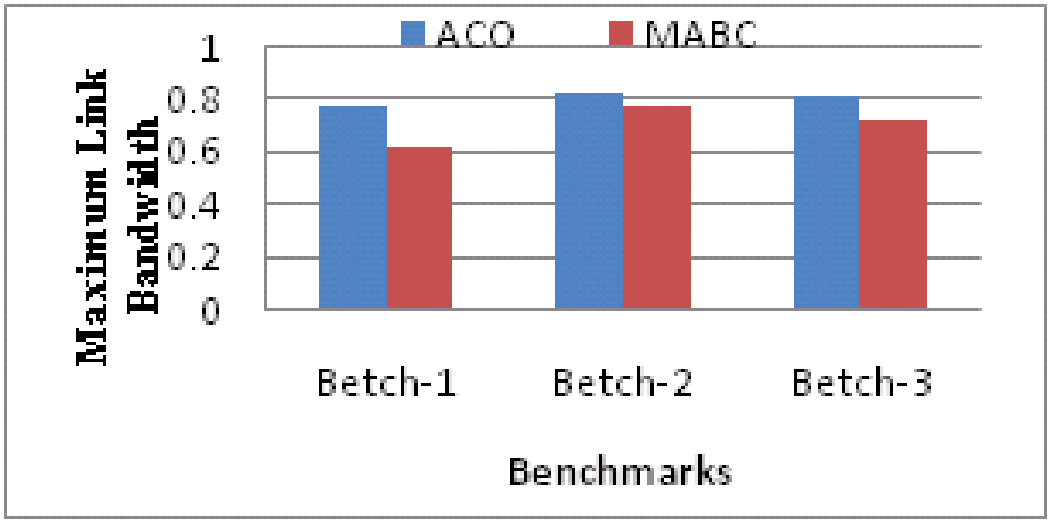

Figure- 2: Maximum Link Bandwidth comparisons for three random benchmarks

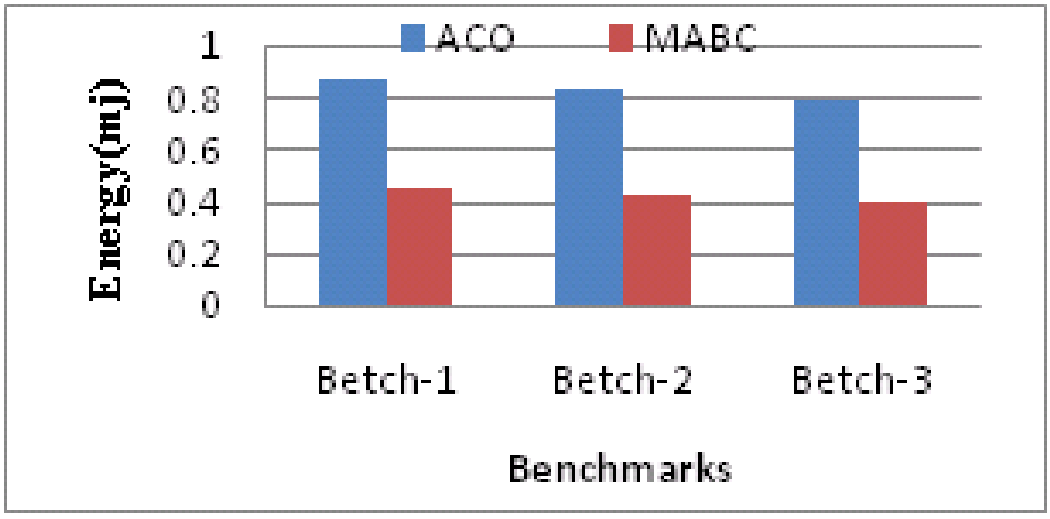

Figure- 3: Energy comparisons for three random benchmarks

Conclusion

In this paper, a multi-objective ABC based heuristic algorithm for the design of link-load balance and low energy mesh based NoC architecture. The proposed algorithm optimizes the NoC energy consumption and maximum link bandwidth. The performance of the algorithm is evaluated in comparison with ACO based mapping algorithm. The experimental results indicate that the proposed ABC algorithm is a viable alternative for solving the mapping problems for NoC.

Reference

[1] L. Benini and G. D. Micheli, Networks on chips: a new soc paradigm, Computer (2002), Vol.35, pp.70-78.

[2] Jian WANG, Yubai LI, Song CHAI, Qicong PENG, Bandwidth-Aware Application Mapping for NoC-Based MPSoCs, Journal of Computational Information Systems 7:1 (2011) 152-159

[3] S. Kumar, A. Jantsch, J. P. Soininen, et al, A network on chip architecture and design methodology. In Proc. IEEE Symposium on VLSI(2002), Pittsburgh, USA, pp.117-124, April. 25-26.

[4] T. Bjerregaard and S. Mahadevan, A survey of research and practices of Network-on-chip, ACM Comput. Surv38, 1, Article 1, Jun. 2006.

[5] J. Hu and R. Marculescu, Energy- and performance-aware mapping for regular NoC architectures, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, 24(4), (2005).

[6] S. Murali and G. De Micheli, Bandwidth-constrained mapping of cores onto NoC architectures, *in Proc. DATE (*2004).

[7] U. Y. Ogras and R. Marculescu, It's a small world after all': NoC performance optimization via long-range link insertion, IEEE Trans. on VLSI,14(7), (2006).

[8] W. J. Dally ,B. Towles, Route Packets, Not Wires:On-Chip Interconnection Networks, Proc of DAC'01, New York, USA, 2001, pp. 684–689.

[9] S. Kumar, A. Jantsch, A Network on Chip Architecture and Design Methodology, Proc of VLSI'02, Pittsburgh, Germany, 4, 2002, pp. 105–112.

[10] Tang Lei, Shashi Kumar, A Two Genetic Algorithm for Mapping Task Graphs to a Network on Chip Architecture, Proc of DSD'03, Antalya, Turkey, 2003,

pp. 180–187.

[11] K Srinivasan and K. S. Chatha, and G. Konjevod,Linear programming based techniques for synthesis of network-on-chip architectures,IEEE Transactions on VLSI Systems, 14(4), 2006, pp. 407–420

[12] Krishnan Srinivasa, Karam and S. Chatha, ISIS:A Genetic Algorithm. based Technique for Custom On-Chip Interconnection Network Synthesis",Proc of VLSID'05, Kolkata, India, 2005, pp. 623–628

[13] Martin K.-F. and Thomas Hollstein and Heiko Zimmer and Manfred Glesner, Deadlock-Free Routing and Component Placement for Irregular Mesh-based Networks-on-Chip, Proc of ICCAD'05, San Jose, CA, USA, pp.238-245, 2005.

[14] D. Karaboga. An idea based on honey bee swarm for numerical optimization [R]. Tech. Rep. TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[15] D. Karaboga and B. Akay. Artificial Bee Colony(ABC) Algorithm on training artificial neural networks[C]. In Proceedings of the IEEE 15th Signal Processing and Communications Applications (SIU '07), 2007, 318-329.

[16] R. P. Dick, D. L. Rhodes, W. Wolf, TGFF: Task graphs for free, Proc of Int. Workshop Hardware/Software Codesign , Seattle, Washington, pp.97–101, 1998.